

Распределенные алгоритмы и системы

mk.cs.msu.ru → Лекционные курсы → Распределенные алгоритмы и системы

Блок 13

Коммуникационный протокол с таймерами

Лектор:

Подымов Владислав Васильевич

E-mail:

valdus@yandex.ru

Рассматриваемая задача

Усложним задачу передачи данных, рассмотренную на примере симметричного протокола раздвижного окна:

- ▶ Передача данных может занимать длительное время, и следует учитывать установку и завершение соединения (*сеанса связи*) между узлами и его разрыв при длительном обмене данными
- ▶ Разрешим сообщениям не только теряться в канале связи, но и
 - ▶ дублироваться: могут появляться копии сообщений, имеющих в канале
 - ▶ перемешиваться: порядок отправки сообщений может не совпадать с порядком их приёма

Этот расширенный спектр ошибок соответствует сетевому уровню модели OSI

Для решения такой усложнённой задачи передачи данных в 1978 году был предложен протокол Δt транспортного уровня

Рассматриваемая задача

По сравнению с BSWP, рассматриваемый (упрощённый) вариант протокола Δt имеет следующие особенности

Однонаправленность: требуется передать данные в одну сторону,
 $p \rightarrow q$

Узел p будем называть **отправителем**, а q — **получателем**

Окно приёма ширины 1: получатель в каждый момент времени ожидает блок данных с одним конкретным порядковым номером, блоки с другими номерами игнорируются

Наличие таймеров — устройств измерения физического времени

Подробно рассмотрим только упрощённую версию протокола с небольшим числом таймеров

Таймеры и допущения о времени

Таймером будем называть переменную, значениями которой являются действительные (вещественные) числа (\mathbb{R})

Будем полагать, что система выполняется в условиях глобального (физического, действительного) времени:

- ▶ Глобальное время линейно
- ▶ Каждое действие выполняется мгновенно (имеет длительность 0) в некоторый заданный момент глобального времени
- ▶ Узлы системы не могут обозревать глобальное время

Таймеры и допущения о времени

Будем считать, что по умолчанию (когда не выполняется явно присваивание) значение таймера убывает в темпе глобального времени:

- ▶ $X^{(t)}$ — значение таймера в момент глобального времени t
- ▶ По умолчанию верно $X^{(t_1)} - X^{(t_2)} = t_2 - t_1$

Также будем считать, что каждый пакет имеет **максимальное время жизни** μ :

- ▶ Если пакет был отправлен в момент t_s и получен в момент t_r , то верно $t_s < t_r < t_s + \mu$, иначе пакет теряется
- ▶ Если пакет дублируется, то временем отправки считается время отправки исходного пакета

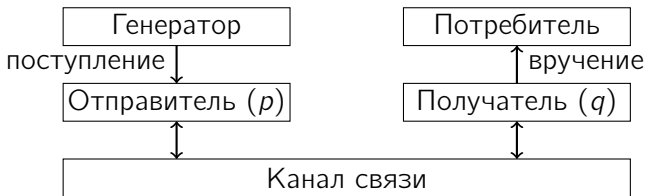
timer — так будем обозначать тип таймера (вещественной переменной)

Другие допущения

Будем считать, что требуется передать от p к q потенциально неограниченный массив блоков данных in_p , недоступный целиком узлу p

Элементы этого массива **поступают** в узел p из **генератора** по порядку следования в массиве при помощи соответствующего действия

Узел q после приёма очередного блока данных d должен **вручить** этот блок **потребителю** выполнением команды $done(d)$



Основная задача узлов p и q — обеспечить *как можно более* надёжную передачу данных от генератора к потребителю

Некоторые блоки данных будут помечаться как «**вероятно потерянные**», и это будет обозначаться значением \updownarrow в особом массиве $lost$

Основные свойства протокола

Отсутствие потерь: для некоторой константы C каждое слово из in_p спустя время C после поступления от генератора будет вручено потребителю или помечено как вероятно потерянное

Полного отсутствия вероятно потерянных блоков избежать невозможно, как бы ни был устроен протокол

Соблюдение порядка: данные вручаются узлом q согласно возрастанию номеров в in_p

Описание протокола: параметры

Будем использовать в протоколе несколько заранее заданных положительных вещественных констант (параметров протокола):

- ▶ μ — максимальное время жизни всех пакетов
- ▶ t — время активности передачи блока
- ▶ r — ожидание после приёма пакета, $r \geq t + \mu$
- ▶ s — ожидание после отправки пакета, $s \geq r + 2\mu$

Описание протокола: переменные отправителя p

Основные переменные:

- ▶ in_p : **array of word** — массив блоков данных для передачи
- ▶ $Low, High$: $\mathbb{N}_0 = 0$
Этими переменными обозначается **окно передачи** $Low \leq i < High$
- ▶ $\vec{\tau}_t$: **array of timer** $= (0, 0, \dots)$ — массив таймеров, отсчитывающих длительность \mathbf{t} с момента получения блока
- ▶ τ_s : **timer** $= 0$ — таймер, отсчитывающий длительность \mathbf{s} с момента последней отправки пакета

Вспомогательные переменные:

- ▶ c_p : **bool** $= \mathbb{f}$ — соединение установлено в узле p
- ▶ N : $\mathbb{N}_0 = 0$ — переменная, отсутствующая в реализации, отмечающая суммарное число сообщений, считающихся врученными или вероятно потерянными и необходимая в обосновании корректности и для удобства использования массивов

Описание протокола: переменные получателя q

Основные переменные:

- ▶ $Exp : \mathbb{N}_0 = 0$ — номер очередного ожидаемого блока
- ▶ $\tau_r : \mathbf{timer} = 0$ — таймер, отсчитывающий длительность \mathbf{r} с момента последнего приёма ожидаемого пакета

Вспомогательные переменные:

- ▶ $c_q : \mathbf{bool} = \mathbb{f}$ — соединение установлено в узле q

Описание протокола: переменные коммуникационной подсистемы

- ▶ M_q — мультимножество **пакетов данных**, адресованных q
Пакет данных имеет вид $(\mathbf{data}, \langle b, i, d, \rho \rangle)$, где:
 - ▶ $b \in \mathbf{bool}$ — **признак начала** окна передачи: индикатор того, что пересылаемый блок данных отвечают наименьшему номеру окна передачи
Этот признак используется получателем, чтобы понять, следует ли установить соединение
 - ▶ $i \in \mathbb{N}_0$ — номер пересылаемого блока данных
 - ▶ $d \in \mathbf{word}$ — пересылаемый блок данных
 - ▶ $\rho \in \mathbb{R}$ — **оставшееся время жизни** пакета: при отправке равно μ , и уменьшается с течением времени (это будет моделироваться соответствующим действием)
- ▶ M_p — мультимножество **пакетов подтверждения**, адресованных p
Пакет подтверждения имеет вид $(\mathbf{ack}, \langle i, \rho \rangle)$, где:
 - ▶ $i \in \mathbb{N}_0$ — номер блока данных, приём которого подтверждается
 - ▶ $\rho \in \mathbb{R}$ — оставшееся время жизни пакета

Описание протокола: схема передачи данных

- p*: Блок d с очередным номером i поступает из генератора
Блок d отправляется
- .. Если пакет данных слишком долго находится в канале, то он удаляется
- q*: Блок d принимается, его номер сверяется с ожидаемым, и при совпадении блок вручается потребителю (а при несовпадении игнорируется)
Если пакет с ожидаемым номером долго не приходит, то выполняется переход к ожиданию следующего блока
Если блок d вручен, то отправляется соответствующее подтверждение
- .. Если подтверждение слишком долго находится в канале, то оно удаляется
- p*: Подтверждение принимается, и выносится решение об успешной передаче блока
Если подтверждение долго не приходит, то блок помечается как возможно потерянный и больше не передаётся

Описание протокола: действия отправителя

В отправителе содержится пять действий (процедур):

1. G_p : поступление очередного блока с возможным установлением соединения
2. S_p : отправка блока
3. R_p : получение подтверждения
4. L_p : пометка блока как возможно потерянного
5. C_p : завершение соединения

Описание протокола: действия отправителя

Поступление блока (G_p)

1. Если $c_p = \mathbb{f}$, то:
 - 1.1 Устанавливается соединение: $c_p := \mathbb{t}$;
 - 1.2 $\tau_s := \mathbf{s}$;
2. $\vec{\tau}_t[N + High] := \mathbf{t}$;
3. $High := High + 1$;

Соединение устанавливается, если оно отсутствует и из генератора поступает очередной блок

Действия в протоколе устроены так, что перед установкой соединения верно $Low = High = 0$

При установке соединения обновляется время ожидания после отправки данных

В любом случае при поступлении блока:

- ▶ Обновляется время продолжения попыток его передать
- ▶ Обновляется правая створка окна передачи (блок сохраняется в p)

Описание протокола: действия отправителя

Отправка блока (S_p)

Предусловие: $c_p \& Low \leq i < High \& \vec{\tau}_t[N + i] > 0$

1. $send(\mathbf{data}, (i = Low), i, in_p[N + i], \mu)$
2. $\tau_s := \mathbf{s};$

Отправить блок можно только в том случае, если

- ▶ соединение установлено,
- ▶ он сохранён в p (принадлежит окну передачи) и
- ▶ время попыток передачи блока ещё не истекло

После отправки пакета обновляется таймер, отсчитывающий соответствующую длительность

Описание протокола: действия отправителя

Приём подтверждения (R_p)

Предусловие: $c_p \&(\mathbf{ack}, \langle i, \rho \rangle) \in M_p$

1. $receive(\mathbf{ack}, \langle i, \rho \rangle)$
2. $Low := \max(Low, i);$

Принять подтверждение можно только в том случае, если

- ▶ соединение установлено и
- ▶ подтверждение есть в коммуникационной подсистеме

После приёма подтверждения обновляется левая створка окна передачи:
 i — номер, следующий за номером успешно принятого пакета

Описание протокола: действия отправителя

Пометка блока как потерянного (L_p)

Предусловие: $c_p \ \& \ \bar{\tau}_t[N + Low] \leq -(\mathbf{r} + 2\mu)$

1. $lost[N + Low] := \mathbb{t}$;
2. $Low := Low + 1$;

Пометить как возможно потерянный можно только наименьший блок окна передачи и только в том случае, если

- ▶ сеанс установлен и
- ▶ с поступления блока прошло хотя бы $\mathbf{t} + \mathbf{r} + 2\mu$ единиц времени — столько требуется суммарно для отправки данных и получения подтверждения в наихудшем (наиболее медленном) случае

После пометки левая створка окна передачи сдвигается: больше этот блок не передаётся

Описание протокола: действия отправителя

Завершение соединения (C_p)

Предусловие: $c_p \ \& \ \tau_s < 0 \ \& \ Low = High$

1. $N := N + High$;
2. $Low := 0$;
3. $High := 0$;
4. Завершить соединение: $c_p := \mathbb{f}$;

Завершить соединение можно только в том случае, если

- ▶ оно установлено,
- ▶ истекло время, выделенное для отправки данных, и
- ▶ окно передачи закрылось

Блоки с номерами меньше N , отвечающими завершённым соединениям, считаются доставленными или вероятно потерянными и удаляются из p . В реализациях протокола нет переменной N , и блоки нумеруются с нуля для каждого соединения.

Пока соединения нет, обеспечивается равенство $Low = High = 0$.

Описание протокола: действия получателя

В получателе содержится два действия (две процедуры):

1. R_q : получение пакета с возможным установлением соединения
2. S_q : отправка подтверждения

Соединение завершается сразу по истечении таймера τ_r , и это обозначено далее в особом действии системы, отвечающем течению времени

Отправка подтверждения (S_q)

Предусловие: c_q

1. $send(\mathbf{ack}, \langle Exp, \mu \rangle)$

Подтверждение отправляется только в том случае, если установлено соединение, и отсылается текущий номер ожидаемого блока (следующий за последним принятым)

Описание протокола: действия получателя

Получение пакета (R_q)

Предусловие: $(\mathbf{data}, \langle b, i, d, \rho \rangle) \in M_q$

1. *receive*($\mathbf{data}, \langle b, i, d, \rho \rangle$)
2. Если $c_q = \mathbb{f}$ & $b = \mathbb{t}$:
 - 2.1 Устанавливается соединение: $c_q := \mathbb{t}$;
 - 2.2 $\tau_r := \mathbf{r}$;
 - 2.3 $Exp := i + 1$;
 - 2.4 *done*(d)
3. Если $c_q = \mathbb{t}$ & $i = Exp$:
 - 3.1 $\tau_r := \mathbf{r}$;
 - 3.2 $Exp := Exp + 1$;
 - 3.3 *done*(d)

Принять пакет можно только в том случае, если он есть в M_q
Если соединения нет и прислан первый блок окна передачи, то устанавливается соединение и принимается блок

Если соединение установлено и прислан блок с ожидаемым номером, то принимается блок

Описание протокола: действия получателя

Получение пакета (R_q)

Предусловие: $(\mathbf{data}, \langle b, i, d, \rho \rangle) \in M_q$

1. *receive*(**data**, $\langle b, i, d, \rho \rangle$)
2. Если $c_q = \mathbb{f}$ & $b = \mathbb{t}$:
 - 2.1 Устанавливается соединение: $c_q := \mathbb{t}$;
 - 2.2 $\tau_r := \mathbf{r}$;
 - 2.3 $Exp := i + 1$;
 - 2.4 *done*(d)
3. Если $c_q = \mathbb{t}$ & $i = Exp$:
 - 3.1 $\tau_r := \mathbf{r}$;
 - 3.2 $Exp := Exp + 1$;
 - 3.3 *done*(d)

Приём блока состоит во вручении блока потребителю, перехода к ожиданию следующего блока и (пере)установке таймера, отсчитывающего время до закрытия соединения

Описание протокола: особые действия

Возникновение ошибок в ненадёжном канале связи будем считать действиями особого узла Ω , вносящего эти ошибки

Кроме того, как действие узла Ω представим и течение времени

Таким образом, в узле Ω содержится три действия:

1. **Loss**: потеря сообщения
2. **Dup**: дублирование сообщения
3. **Time**: течение времени

Потеря сообщения (Loss)

Предусловие: $m \in M_x$ & $x \in \{p, q\}$

1. $M_x := M_x - \{m\};$

Дублирование сообщения (Dup)

Предусловие: $m \in M_x$ & $x \in \{p, q\}$

1. $M_x := M_x + \{m\};$

Описание протокола: особые действия

Течение времени (Time)

1. Выбрать произвольное значение $\Delta \in \mathbb{R}, \Delta > 0$
2. Для каждого $i \in \mathbb{N}_0$: $\vec{\tau}_s[i] := \vec{\tau}_s[i] - \Delta$;
3. $\tau_s := \tau_s - \Delta$;
4. $\tau_r := \tau_r - \Delta$;
5. Если $\tau_r \leq 0$:
 - 5.1 Завершить соединение в q : $c_q := \mathbb{f}$;
6. Для каждого пакета $m = (\dots, \langle \dots, \rho \rangle) \in M_x, x \in \{p, q\}$:
 - 6.1 $\rho := \rho - \Delta$;
 - 6.2 Если $\rho \leq 0$:
 - 6.2.1 $M_x := M_x - \{m\}$;

При выполнении этого действия значения всех таймеров системы уменьшаются на одну и ту же положительную константу Δ

Если время ожидания в q истекло, то соединение завершается

Если время жизни пакета истекло, то он удаляется